

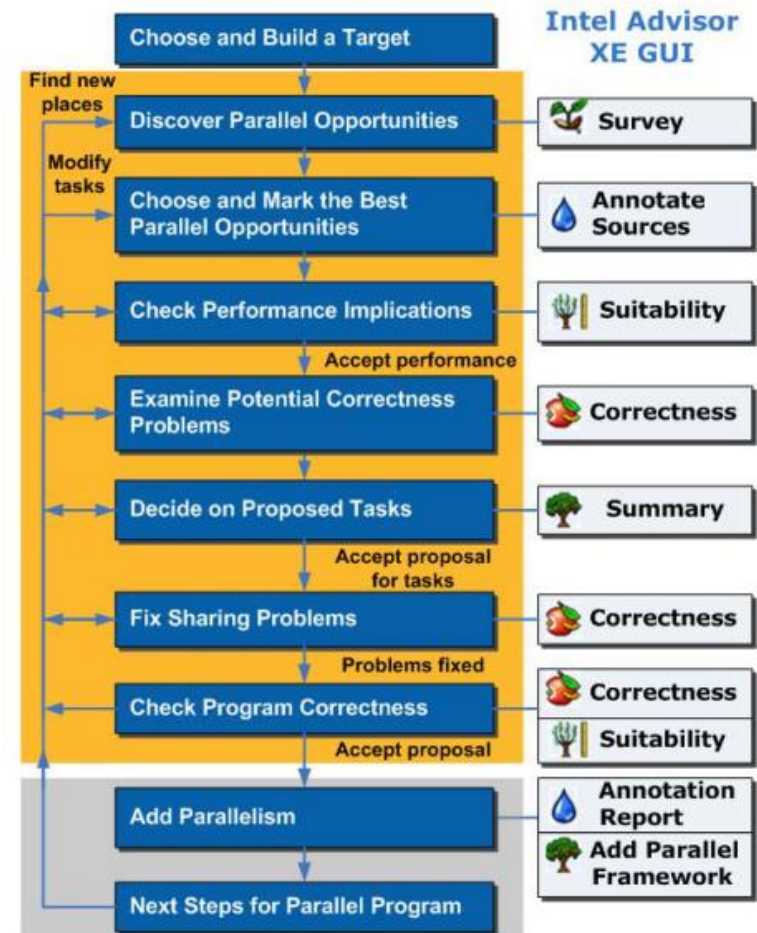


Intel® Advisor XE 2013



Intel® Advisor XE

- Tool for what-if analysis
 - Modeling: use code annotations to introduce parallelism
 - Evaluation: estimate the effect e.g. the speedup
 - GUI-driven assistant (5 steps)
- Productivity and Safety
 - Parallel correctness is checked based on a correct program
 - Non-intrusive API
- It's not auto-parallelization
- It's not modifying the code














Intel® Advisor XE

Stages

Transforming many serial algorithms into parallel form takes 5 easy high-level steps:

1. **Survey and Summary tools:** where to add parallelism
2. **Annotations:** experiment with parallel program structure
3. **Suitability tool:** predict and model program scalability
4. **Correctness tool:** discover potential synchronization problems
5. **Manually convert annotations to parallel framework API** (with a little help of Annotations/Summary)

Advisor XE Workflow

- 1. Survey Target**
[Where](#) should I consider adding parallelism? Locate the loops and functions where your program spends its time, and functions that call them.
  Collect Survey Data
 View Survey Result
- 2. Annotate Sources**
Add Intel Advisor XE annotations to [identify](#) possible parallel tasks and their enclosing parallel sites.
+ Steps to annotate
 View Annotations
- 3. Check Suitability**
Analyze the annotated program to check its predicted parallel [performance](#).
  Collect Suitability Data
 View Suitability Result
- 4. Check Correctness**
[Predict](#) parallel data sharing problems for the annotated tasks. [Fix](#) the reported sharing problems.
  Collect Correctness Data
 View Correctness Result
- 5. Add Parallel Framework**
+ Steps to replace annotations
 View Summary

Current Project: Benchmarks

Survey Target

The screenshot displays the Intel Advisor XE Workflow interface. The left sidebar contains a navigation pane with four steps: 1. Survey Target, 2. Annotate Sources, 3. Check Suitability, and 4. Check Correctness. The main window shows the 'Summary of predicted parallel behavior' for the project '1_tachyon_serial - e000'. The summary includes a message that no annotations were found in the source files and a table of top time-consuming loops. The bottom of the interface shows the 'Output' pane with the message 'Show output from: Intel Advisor XE 2013 messages'.

Advisor XE Workflow 1_tachyon_serial - e000 Conversion Report Conversion Report Intel Advisor XE 2013

1. Survey Target
Where should I consider adding parallelism? Locate the loops and functions [\[read more\]](#)
Collect Survey Data
View Survey Result

2. Annotate Sources
Add Intel Advisor XE annotations to [identify](#) possible parallel tasks and their enclosing parallel sites.
Steps to annotate
View Annotations

3. Check Suitability
Analyze the annotated program to check its predicted parallel [performance](#).
Collect Suitability Data
View Suitability Result

4. Check Correctness
[Predict](#) parallel data sharing problems for the annotated tasks. [Fix](#) the reported sharing problems.
Collect Correctness Data

Summary of predicted parallel behavior

Summary Survey Report Annotation Report Suitability Report Correctness Report

Intel Advisor XE helps you choose where to add parallelism to your program
Intel Advisor XE tools help you choose possible parallel code regions, and predict their approximate parallel performance and data sharing problems. View the Advisor XE Workflow to guide you.

No Annotations found in your project source files.
After scanning 79 source files, 0 annotations have been found.

Top time-consuming loops[®]
Consider adding parallel site and task annotations around these time-consuming loops found during Survey analysis.

Loop	Source Location	CPU Total Time [®]
parallel thread	tachyon_serial.cpp:158	20.9475s
parallel thread	tachyon_serial.cpp:167	20.9172s
shader	shade.cpp:104	8.0193s
intersect_objects	intersect.cpp:107	7.9018s
grid_intersect	grid.cpp:550	7.4965s

Collection Details

Survey
Collection started: 10 October 2013, 14:12:55
Collection finished: 10 October 2013, 14:13:28

Output
Show output from: Intel Advisor XE 2013 messages

Current Project: 1_tachyon_serial

Annotate Sources
















What are annotations?

- Macros (function calls) that can be easily disabled*
- Inform analysis tool about intended parallelization

Example Intel Advisor annotations:

```
#define ANNOTATE_SITE_BEGIN(NAME)      call annotate_site_begin(NAME)
#define ANNOTATE_SITE_END(NAME)        call annotate_site_end()
#define ANNOTATE_ITERATION_TASK(NAME)  call annotate_iteration_task(NAME)
#define ANNOTATE_TASK_BEGIN(NAME)      call annotate_task_begin(NAME)
#define ANNOTATE_TASK_END(NAME)        call annotate_task_end()
#define ANNOTATE_LOCK_ACQUIRE(ADDRESS) call
annotate_lock_acquire(ADDRESS)
#define ANNOTATE_LOCK_RELEASE(ADDRESS) call
annotate_lock_release(ADDRESS)
```

Annotate Sources using Survey Data

Function Call Sites and Loops	Total Time %	Total Time
▢ rt_renderscene	96.2% 	19.9664s
▢ renderscene	96.2% 	19.9662s
▢ trace_region	96.2% 	19.9662s
▢ trace_shm	96.2% 	19.9662s
▢ thread_trace	96.2% 	19.9662s
▢  parallel_thread [loop]	96.2% 	19.9662s
▢  parallel_thread [loop]	96.1% 	19.9462s
▢ parallel_thread	96.0% 	19.9262s
▢ render_one_pixel	93.1% 	19.3208s
▢ trace	79.2% 	16.4369s
▢ shader	43.6% 	9.0521s
▢  shader [loop]	35.4% 	7.3348s

**Pick out expensive loop
and add annotations**

```

ANNOTATE_SITE_BEGIN(allRows);
for (int y = starty; y < stopy; y++)
{
    ANNOTATE_TASK_BEGIN(eachRow);
    m_storage.serial = 1;
    m_storage.mboxsize = sizeof(unsigned);
    m_storage.local_mbox = (unsigned*)
    memset(m_storage.local_mbox, 0, m

    drawing_area drawing(startx, tot
    for (int x = startx; x < stopx;
        color_t c = render_one_pixel
        drawing.put_pixel(c);
    }

    if(!video->next_frame())
    {
        free(m_storage.local_mbox);
        return;
    }
    free(m_storage.local_mbox);
    ANNOTATE_TASK_END(eachRow);
}
ANNOTATE_SITE_END(allRows);

```

Check Suitability

Advisor XE Workflow

2. Annotate Sources

Add Intel Advisor XE annotations to [identify](#) possible parallel tasks and their enclosing parallel sites.

Steps to annotate

View Annotations

3. Check Suitability

Analyze the annotated program to check its predicted parallel performance.

Collect Suitability Data

View Suitability Result

4. Check Correctness

Predict parallel data sharing problems for the annotated tasks. [Fix](#) the reported sharing problems.

Collect Correctness Data

View Correctness Result

5. Add Parallel Framework

Steps to replace annotations

View Summary

Current Project: 1_tachyon_serial

tachyon_serial.cpp

1_tachyon_serial - e000

Conversion Report

What are the performance implications of the annotated sites?

Intel Advisor XE 2013

Summary

Survey Report

Annotation Report

Suitability Report

Correctness Report

All Sites

Maximum Program Gain For All Sites:

2.40x

Target CPU Count: 8

Threading Model: Intel TBB

Annotation ...	Source Location	Maximum Sit ...	Maximum Tot ...	Average Instan ...	Total Ti ...
allRows	tachyon_seri ...	7.92x	2.40x	20.2218s	20.2218s

Selected Site

Scalability of Maximum Site Gain

Target CPU Count	Maximum Site Gain
2	1x
4	2x
8	2.40x
16	2.40x
32	2.40x

Changes I will make to this site to improve performance

Type of Change	Benefit if Checked	Loss if Unchecked	Recommended
<input type="checkbox"/> Reduce Site Overhead			No
<input type="checkbox"/> Reduce Task Overhead	0.02x		No
<input type="checkbox"/> Reduce Lock Overhead			No
<input type="checkbox"/> Reduce Lock Contention			No
<input type="checkbox"/> Enable Task Chunking	0.02x		No

Annotation	Annotation La ...	Source Location	Number of Instan ...	Maximum Instance T ...	Average Instance Ti ...	Minimum Instance Ti ...	Total Time
Selected S ...	allRows	tachyon_serial.cp ...	1	20.2218s	20.2218s	20.2218s	20.2218s

Output

Show output from: Intel Advisor XE 2013 messages

Check Correctness

Advisor XE Workflow

1. Survey Target

Where should I consider adding parallelism? Locate the loops and functions [\[read more\]](#)

Collect Survey Data

View Survey Result

2. Annotate Sources

Add Intel Advisor XE annotations to [identify](#) possible parallel tasks and their enclosing parallel sites.

Steps to annotate

View Annotations

3. Check Suitability

Analyze the annotated program to check its predicted parallel [performance](#).

Collect Suitability Data

View Suitability Result

4. Check Correctness

[Predict](#) parallel data sharing problems for the annotated tasks. [Fix](#) the reported sharing problems.

Collect Correctness Data

Current Project: 1_tachyon_serial

tachyon_serial.cpp

1_tachyon_serial - e000

Conversion Report

Did the annotated tasks expose data sharing problems?

Summary

Survey Report

Annotation Report

Suitability Report

Correctness Report

Problems and Messages

ID	Type	Site Name	Sources	Modules	State
P1	Parallel site information	allRows	tachyon_serial.cpp	1_tachyon_serial.exe	✓ Not a problem
P2	Data communication	allRows	tachyon_serial.cpp; winvideo.h	1_tachyon_serial.exe	New
P3	Memory reuse	allRows	tachyon_serial.cpp	1_tachyon_serial.exe	New
P4	Memory reuse	allRows	tachyon_serial.cpp	1_tachyon_serial.exe	New
P5	Memory reuse	allRows	tachyon_serial.cpp	1_tachyon_serial.exe	New

Memory reuse: Code Locations

ID	Description	Source	Function	Module	State
X5	Parallel site	tachyon_serial.cpp:157	parallel_thread	1_tachyon_serial.exe	New
<pre> 155 //ADVISOR COMMENT: Don't forget to uncomment the #include <advisor-annotate. 156 157 ANNOTATE_SITE_BEGIN(allRows); 158 for (int y = starty; y < stopy; y++) 159 { </pre>					

ID	Description	Source	Function	Module	State
X6	Write	tachyon_serial.cpp:161	parallel_thread	1_tachyon_serial.exe	New
<pre> 159 { 160 ANNOTATE_TASK_BEGIN(eachRow); 161 m_storage.serial = 1; </pre>					

Filter

Severity

Error 4

Remark 1 item

Type

Parallel site inf... 1 item

Data commun... 1 item

Memory reuse 3

Site Name

allRows 5

Source

tachyon_serial.cpp 5

winvideo.h 1 item

Module

1_tachyon_serial.exe 5

State

New 4

Sort By Item Name

Output

Show output from: Intel Advisor XE 2013 messages

Add Parallel Framework

The screenshot displays the Intel Advisor XE 2013 interface. On the left, a sidebar titled 'Advisor XE Workflow' shows five steps: 2. Annotate Sources, 3. Check Suitability, 4. Check Correctness, and 5. Add Parallel Framework. Step 5 is highlighted with a blue border. The main window shows the 'Summary of predicted parallel behavior' for 'tachyon_serial.cpp'. It includes a 'Summary' tab and a 'Collection Details' section with a table of parallel sites and their performance metrics.

Advisor XE Workflow

- 2. Annotate Sources
 - Add Intel Advisor XE annotations to [identify](#) possible parallel tasks and their enclosing parallel sites.
 - Steps to annotate
 - View Annotations
- 3. Check Suitability
 - Analyze the annotated program to check its predicted parallel [performance](#).
 - Collect Suitability Data
 - View Suitability Result
- 4. Check Correctness
 - [Predict](#) parallel data sharing problems for the annotated tasks. [Fix](#) the reported sharing problems.
 - Collect Correctness Data
 - View Correctness Result
- 5. Add Parallel Framework
 - Steps to replace annotations
 - View Summary

Summary of predicted parallel behavior Intel Advisor XE 2013

Summary Survey Report Annotation Report Suitability Report Correctness Report

Intel Advisor XE helps you choose where to add parallelism to your program

Intel Advisor XE tools help you choose possible parallel code regions, and predict their approximate parallel performance and data sharing problems. View the Advisor XE Workflow to guide you.

Annotations found in your project source files.

After scanning 79 source files, 5 annotations have been found.

Potential program gain[®]: 2.40x (8 CPUs, Intel TBB Threading Model)

These annotated parallel sites were detected:

Parallel Site	Maximum Site Gain [®]	Correctness Problems
allRows (tachyon_serial.cpp:157)	7.92x	4 0

Consider adding parallel site and task annotations around these time-consuming loops found during Survey analysis.

Loop	Source Location	CPU Total Time [®]
parallel thread	tachyon_serial.cpp:158	19.9662s
parallel thread	tachyon_serial.cpp:167	19.9462s
shader	shade.cpp:104	7.3348s
intersect objects	intersect.cpp:107	7.2290s
grid_intersect	grid.cpp:550	6.8997s

Collection Details

Output

Show output from: Intel Advisor XE 2013 messages

Add Parallel Framework

How can Intel Advisor help?

- Contains overhead metrics for selected models
- Allows to play with some implementation options
- Detailed help pages for different frameworks.

Serial Code with Intel Advisor Annotations	Parallel Code using Intel TBB
<pre>// Locking ANNOTATE_LOCK_ACQUIRE(); Body(); ANNOTATE_LOCK_RELEASE();</pre>	<pre>// Locking can use various mutex types provided // by Intel TBB. For example: #include <tbb/tbb.h> ... tbb::mutex g_Mutex; ... { tbb::mutex::scoped_lock lock(g_Mutex); Body(); }</pre>
<pre>// Do-All Counted loops, one task ANNOTATE_SITE_BEGIN(site); For (I = 0; I < N; ++I) { ANNOTATE_ITERATION_TASK(task); {statement;} } ANNOTATE_SITE_END();</pre>	<pre>// Do-All Counted loops, using lambda // expressions #include <tbb/tbb.h> ... tbb::parallel_for(0,N,[&](int I) { statement; });</pre>

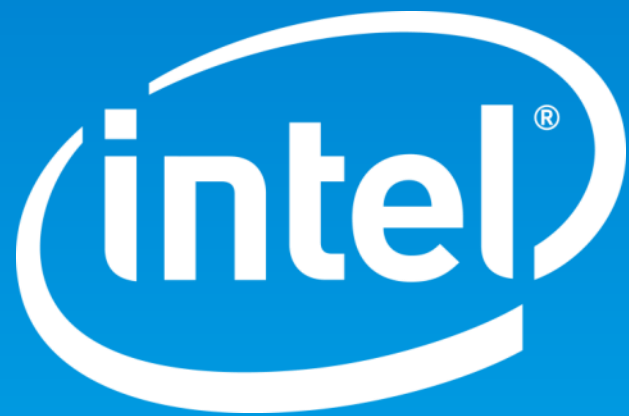
In Summary

Advisor XE is a tool to assist in the parallelization of serial code.

It helps the developer/architect by

- Finding the sections of code taking up most of the time
- Modeling the available speedup for parallelizing those sections of code.
- Finding correctness errors that will occur if the section is naively parallelized

It does all this using a lightweight annotation system to allow quickly trying multiple code change possibilities without choosing a parallel framework.



Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © 2014, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

